**1. Grundsätzlich für WPF**

* **.NET 6/7/8 (LTS bevorzugen)** → Basisframework, modern, gute Performance
* **MVVM-Frameworks** für saubere Architektur:
  + Prism → sehr verbreitet, bietet Navigation, Commands, Dependency Injection
  + CommunityToolkit.Mvvm (ehem. MVVM Toolkit) → leichtgewichtig, Microsoft-offiziell, einfach zu lernen
  + ReactiveUI → für reaktive Programmierung (falls ihr Events/Asynchronität stark nutzt)

Empfehlung für euch: **CommunityToolkit.Mvvm**, weil es einfach ist und wenig Overhead hat.

**2. GUI & Styling**

* **MahApps.Metro** → modernes UI-Design für WPF, Metro-Stil, Themes
* **MaterialDesignInXAML** → Google Material Design für WPF
* **LiveCharts2** → Diagramme (Fehlerstatistiken, Kreis-/Balkendiagramme)
* **OxyPlot** → Alternative zu LiveCharts, simpler für wissenschaftliche Diagramme

**3. Datenbank & Datenzugriff**

* **Entity Framework Core** → ORM für SQLite / SQL Server Express (sauberer Zugriff, LINQ-Abfragen)
* **Dapper** → leichtgewichtige Alternative, schneller, aber weniger Komfort

Für Ausbildungsprojekt: **Entity Framework Core mit SQLite** → einfach + viele Tutorials.

**4. Datei- und Log-Verarbeitung**

* Reguläre Ausdrücke (Regex, in .NET eingebaut) → perfekt für Logfile Parsing
* Falls ihr sehr große Logfiles habt → **System.IO.Pipelines** für performantes Streamen

**5. Reporting & Export**

* **iText7 / iTextSharp** → PDFs erstellen
* **ClosedXML** → Excel-Export (falls später gewünscht)
* **CsvHelper** → CSV-Dateien lesen/schreiben

**6. Sicherheit**

* **Microsoft.Data.Sqlite + Verschlüsselung** (z. B. über passwortgeschützte DB-Dateien oder zusätzliche Verschlüsselungslogik)
* **BCrypt.Net-Next** → für Passwort-Hashing

**7. Optional (später interessant)**

* **MailKit** → für E-Mail-Benachrichtigungen
* **Quartz.NET** → falls ihr später geplante Jobs/Automatisierungen braucht (z. B. Logs regelmäßig scannen)
* **AutoMapper** → falls die Datenmodelle komplexer werden (Mapping zwischen DB und UI-Objekten)